A Framework for MLS Interoperability

Myong H. Kang , Judith N. Froscher, and Ira S. Moskowitz
Information Technology Division, Mail Code 5540
Center for High Assurance Computer Systems
Naval Research Laboratory
Washington, D.C. 20375, USA
e-mail: mkang@itd.nrl.navy.mil

Abstract
Distributed object-oriented computing (DOC) is a new computing paradigm that promotes component-based development, location independence, scalability, software reuse, etc. Users of multilevel security (MLS) technology want to take advantage of these new technologies. However, the process of incorporating new technologies into MLS products is slower than the analogous process for non-secure commercial products because MLS products must go through rigorous evaluation/certification procedures.

We propose an architectural framework that speeds up the process of introducing new technologies to MLS users. We examine the drawbacks of traditional MLS approaches and take a fresh look at the requirements of MLS users. We then introduce security-critical components that can enable MLS solutions and an MLS architectural framework that can accommodate not only legacy systems but also new technologies, including DOC, without jeopardizing system security. Our framework separates security critical components/functions from the rest of the system because these components must go through rigorous evaluation/certification processes. This approach enables the secure use of new technologies for MLS users.

1. Introduction

The trend of computing today is toward open[1], distributed, object-oriented computing (DOC). CORBA[2] and OLE/COM[3] are evolving standards for these trends. This new computing paradigm is motivated by component based development, location independence, scalability and fault-tolerance, software reuse, etc. In this paradigm, clients send requests to servers and servers return results. Clients know neither where a particular server is located nor which server is serving the request. A broker mediates all accesses (e.g., object request broker in CORBA and COM in OLE/COM). The broker receives requests from the client, finds the object implementation for the request, transmits the request to a server, and returns the output to the client.

Even though DOC promotes desirable features such as interoperability among heterogeneous systems, wrapping legacy applications with objects, and so on, it creates new problems which must be addressed. Some important concerns in DOC environments are:

1. authentication and access control of clients to server objects,
2. privacy, integrity and authenticity of messages in the underlying network, and
3. availability/fault-tolerance.

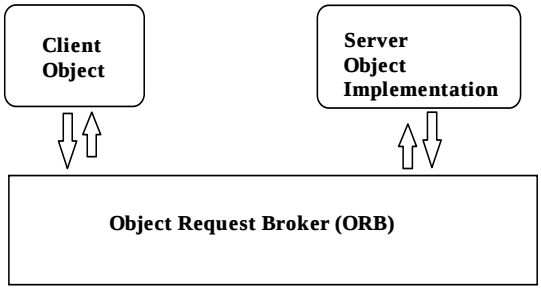

Figure 1: The architecture of a CORBA ORB

The multilevel security (MLS) community has rarely addressed the third concern, the first two concerns have been the main topics of the MLS community for the past 20 years. Therefore, the DOC community can use the techniques developed by the security community for the solutions to concerns (1) and (2). Concern (3) is beyond the scope of this paper.

Enterprises, including the Department of Defense, have certain information to which access must be restricted. The enterprises determine levels of trustworthiness (clearances) for their users. Information is marked with a sensitivity level. By comparing a user's level of trustworthiness with the sensitivity level of information, either a manual or automated procedure can make access control decisions. In an automated system, the mechanisms responsible for making access control decisions must be trusted and are extremely difficult and expensive to develop, evaluate, and certify. MLS systems have a small amount of trusted software that protects the system and the data. The untrusted software in an MLS system may contain malicious code, which can exploit vulnerabilities in the trusted software to leak information. In DOC, such vulnerabilities not only threaten information on a given system, but put all high level information in the confederation at risk. Hence, strong separation (e.g., physical separation or very high assurance separation) of data is essential to enable interoperability among users while protecting sensitive information. Users must be able to ensure security when accessing the information they need to do their jobs, while also taking advantage of the rapid advances made in information technology. Hence, there is a growing need for finding MLS solutions for DOC.

[1] "Open" simply means that the interfaces/protocols are published.
[2] CORBA (Common Object Request Broker Architecture) is from the Object Management Group.
[3] OLE/COM (Object Linking and Embedding/Component Object Model) is from Microsoft Inc.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
1996		**00-00-1996 to 00-00-1996**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
A Framework for MLS Interoperability	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Research Laboratory,Center for High Assurance Computer Systems,4555 Overlook Avenue, SW,Washington,DC,20375**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**9**	

To date, MLS products typically lag new technology because security vendors must develop MLS versions of the new technology, and must convince independent third parties that the protection features are effective. The MLS versions of new technology often trade compatibility, performance, and capability for protection. However, users who must protect classified data also want to take advantage of the full functionality of the new technology as soon as possible.

In this paper, we explain what problems result from the naive application of traditional MLS technology to the DOC environment, and we propose an alternative approach that can speed up the process of introducing new technologies to MLS users and promote information sharing.

Note that the current security work of the DOC community is mainly concentrated on network security and access control mechanisms within a single security level [19]. Here, however, we present a framework of multilevel security, which requires strong separation, that can incorporate the security work of the DOC community.

2. Difficulties with the Naive Extension of Traditional MLS Approaches to Distributed Environments

Traditionally, the MLS community has concentrated on developing computer systems based upon a trusted computing base (TCB). In general, TCBs have a mandatory access control mechanism that is based on the Bell-LaPadula policy (BLP) [1]. BLP is a specific access control policy that is based on a widely accepted information security policy which can be summarized as "no high level information is allowed to pass to lower level users/processes and lower level information should be available to higher level users/processes."

Since user requirements in MLS computing environments hardly differ from those for non-MLS computing environments, future MLS computing environments will be distributed and client-server architecture based. The need for information sharing will be greater in the future. Recently, the composition of MLS systems was investigated to address the problems of distributed MLS computing [17]. In the following, we examine difficulties of a naive extension of the traditional MLS approaches to distributed environments.

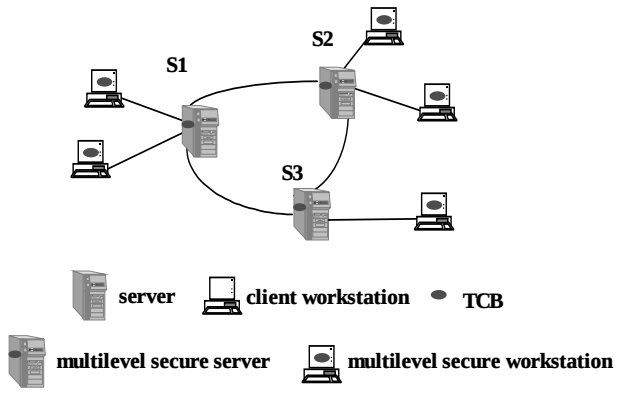

Figure 2: A TCB-based distributed system

2.1. General difficulties

Figure 2 shows an architecture that connects MLS systems to form a distributed MLS system. This architecture shows the naive extension. By this we mean that security level *i* in system *j* communicates only to level *i* in system *j'*. As we can see, TCBs reside on every system. In figure 2, a server may be a computer system or represent a cluster of servers for an organization.

Let us examine some problems associated with such a naive extension of the conventional MLS computing paradigm to client-server based distributed environments.

2.1.1. Assurance class of the whole system

If TCBs from different classes of assurance (e.g., B1, B2, etc. of TCSEC [4]) are naively composed to build a distributed system, the assurance class of the whole system can be no greater than the assurance class of the lowest among the TCBs. This is true because the vulnerability of the whole system is determined by the most vulnerable part of the system.

Therefore, when MLS systems are naively connected to compose an MLS distributed system, the effect of each connected system to overall composition has to be considered carefully. This hampers the interoperability efforts significantly because an organization which has higher assurance (hence, usually expensive) computer systems may be reluctant to connect to another organization that has lower assurance computer systems. In the naive extension, effective protection of secret information must be traded against interoperability.

2.1.2. Migration of legacy systems

There are many existing system-high[4] computer systems (i.e., these systems are not TCB based and may be out-dated). As the computing paradigm changes and the need to share information increases, these legacy systems need to be connected to other systems without violating the security policy of the federation/composition.

The naive composition of MLS systems that was depicted in figure 2 does not provide a simple way for a legacy system to be connected to and to share information with other systems. Therefore, it is desirable to come up with a distributed MLS architecture that can provide a way to incorporate legacy systems.

2.1.3. Migration to new technologies

Computer (hardware and software) technologies are changing rapidly. In general, MLS systems are much slower than non-MLS systems to adopt new technologies because MLS systems have to go through rigorous (thus long) development, evaluation and certification processes. Potential users of MLS technology are reluctant to use old technology for the sake of MLS. Therefore, MLS research should promote/allow using non-MLS components securely as much as possible so that when a new technology is available, non-MLS components can be replaced without jeopardizing system security.

[4] System-high computer systems are non-MLS (i.e., single level) systems that contain data at all security levels up to and including that level. All personnel who access the system have a clearance and all computer facilities are protected according to the requirements for the highest classification of material contained in the system.

2.1.4. Application programs

The number of application programs running on MLS platforms is much smaller than that of application programs running on non-MLS platforms. Hence, the choice of application programs for MLS users is limited. Also, there are many application programs that run on legacy systems. Those programs may have to go through extensive modification to run on MLS systems. This is another reason for an MLS architecture that allows as much use of non-MLS components as possible.

2.1.5. Cost

In general, TCB based systems are much more expensive than non-MLS equivalent systems. Replacing all existing computer systems with TCB based systems is not practical. Therefore, an MLS architecture that allows as many non-MLS products as possible is also cost-efficient. Such an architecture should be able to assign MLS and other critical functions to as few components as possible. In this case only the critical components need to go through a rigorous evaluation process, thus reducing the overall cost.

2.2. Technical difficulties

As we mentioned, BLP is a specific interpretation of a widely-accepted information security policy. BLP is expressed in terms of subjects and objects (to avoid confusion, we call Bell-LaPadula subjects, "BLS", and Bell-LaPadula objects, "BLO", for the rest of the paper). A BLO is a passive entity such as a file or segment that contains or receives information. A BLS is an active entity such as a user or a process, that causes changes to system states or to BLOs. A level is associated with each BLS or BLO. BLP expresses a dominance[5] ($\geq$) relationship between BLS and BLO is as follows:

1. **Simple security property:** A BLS has read access to a BLO only if the security level of the BLS dominates the security level of the BLO.
2. **★-property (Star property):** A BLS has write access to a BLO only if the security level of the BLS is dominated by the security level of the BLO.

Note that BLSs and BLOs are static entities (i.e., in general, a BLS is always a subject and a BLO is always an object). However, in the DOC paradigm, there are client objects and server objects. A client object sends a message to server objects to receive services, and a server object returns answers to the client object. Also, the concept of client and server is not static (i.e., objects can alternate between client and server roles).

BLP was well suited for monolithic MLS systems that focused on system level security. However a direct extension of BLP to the DOC paradigm has some difficulties. Hence, we may need to go back to the original security policy underlying BLP that is based on information flow and re-examine the security of the entire system.

The naive extension of the traditional MLS approach to distributed systems has another difficulty. BLP allows a BLS from a higher level to access information from lower level BLOs. Since, under BLP, a higher level BLS cannot send requests to lower level BLOs, the underlying TCB provides read-down capability that allows it to share lower level information with a higher level BLS. However, in the DOC environments, the lower level information may be located in separate computers which have either different TCBs that control access to different ranges of security levels, or operating systems that may not even have a TCB (see figure 3 where security levels are L1 < L2 < L3 < L4 and dashed lines represent desired connections).

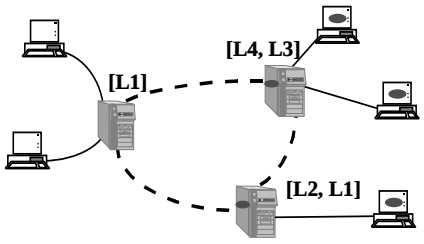

Figure 3: Mixture of MLS and non-MLS systems

Since read-down from one system to another is not guaranteed, and sending requests to lower level systems is prohibited, new ways to share information are needed for new MLS computing paradigms.

One may argue that an MLS object request broker (ORB) solves the problem. Some difficulties with the MLS ORB approach are as follows:

- High assurance MLS ORB may not be practical because the customer base of MLS technology is not as broad as that of non-secure technology.
- Even if a high assurance ORB would be developed, sending requests from higher level clients to lower level servers introduces security vulnerabilities (i.e., covert information can be hidden in legitimate requests). A similar approach has been tested and abandoned for security reasons [8].

In the following section, we examine a few typical MLS distributed configurations and derive functional requirements based on the information security policy.

3. Key Configurations of MLS Distributed Systems

There are compelling operational reasons for sharing information among computer systems from different security levels without violating the security policy of the federation. These requirements force many systems (including system-high and legacy systems) to be connected to a network. In this section, we consider possible configurations and derive functional requirements. Critical components that can meet the functional requirements are discussed in the following section. As a convention, we use L1, L2, L3, and L4 as security levels where L1 < L2 < L3 < L4.

3.1. Configuration 1

We show the simplest configuration for connecting systems from two different levels in figure 4.

[5] Level L2 is said to dominate (strictly dominate) level L1, L2 ≥ L1 (L2 > L1), if the hierarchical classification of L2 is greater than or equal (and not equal) to that of L1 and non-hierarchical categories of L2 include all those of L1 as a subset.

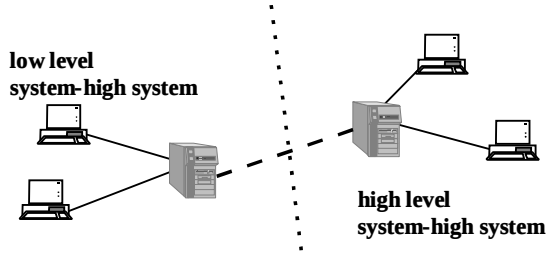

Figure 4. Two system-high systems need to be connected

In this configuration, a low level system-high system and a high level system-high system are connected through a network. The regular dotted line in figure 4 represents a logical security boundary between two levels and the thick dashed line represents a needed connection. An application of this type of architecture to the Joint Maritime Command Information System, an integrated C^4I system, is described in detail in [5].

3.2. Configuration 2

The cost of building a large long-haul network (e.g., Internet) remains high, which encourages the sharing of network resources. Some organizations may have their dedicated system-high LAN already. Other organizations may already have an MLS LAN. These organizations may also want to connect their LAN to a long-haul network to access information. Sometimes higher level users may need to access lower level computing resources (e.g., information, hardware). Figure 5 shows such an architecture.

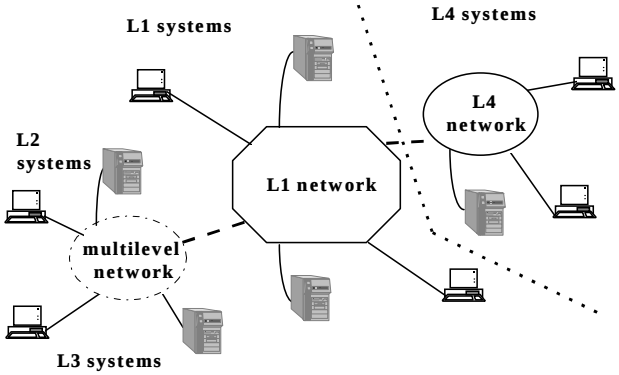

Figure 5: L4 and MLS networks need be connected to a long haul L1 network

3.3. Functional requirements

The question is how to connect systems from different security levels and meet the operational needs without compromising the security of the (federated) system. What are the functional requirements necessary for promoting information sharing in the new architectures? (Of course, higher level information may not be shared with lower level users, but lower level information should be accessible by higher level users.) We list a few functional requirements below.

- ***Information flow from low to high systems.*** Higher level messages/requests cannot flow to lower level systems. However, lower level information can flow/replicate to higher level systems without violating the security policy. Of course, information that has to be sent from low to high systems has to be pre-arranged between organization/systems. The infrastructure for providing this service must be a part of new MLS architectures. If some information is replicated to higher level systems, the consistency of the replica is an important problem. Details of maintaining consistency of the replica can be found in [9, 20].
- ***Privacy, integrity, authenticity of information.*** In an open distributed computing environment, privacy, integrity, and authenticity of information are important concerns. Cryptographic techniques can be useful in these cases.
- ***Higher level users may need to access lower level resources.*** Recently, large amounts of valuable information have been made available in lower level networks (e.g., Internet web sites). Even though some lower level information can be sent (replicated) to higher level systems, not all necessary information can be replicated to higher level systems (because, sometimes, the usage of lower level information at a higher level system is difficult to predict). Hence, the high level users may need to search through lower level information. Note that the security requirement here is to provide the capability for a high level user to login to lower level services as a low user.
- ***Availability of resources.*** In the DOC environment, malicious/nonmalicious users/processes can monopolize computing resources. Also failure of a single component can hamper a critical operation. Special mechanisms are needed to prevent/alleviate such misuse/failure.
- ***Downgrading.*** Some information from higher level systems may need to be released to lower-level users (e.g., out dated information, after removing critical portions). Note that this requirement violates the security policy.

4. Critical Components

In the previous section, we examined the functional requirements for MLS federations. In this section, we identify some critical components that can help to satisfy the functional requirements. These critical components, in conjunction with some other techniques (e.g., replication), will form the backbone of a new security framework that will be presented in section 5.

The security critical components should be designed independently of any specific computing paradigm so that these components can be used with any existing technology and possibly emerging technologies (i.e., *separation of concerns*). Cryptographic components are well-known examples of such independence. In other words, a cryptographic module does not have any knowledge about incoming data (e.g., object, file, etc.). It simply receives a bit stream and outputs another bit stream. The interpretation and the use of this bit stream is entirely up to the application programs that use a cryptographic module.

Note that there is no one component that can solve the availability/fault-tolerant problem of the system. This problem can be solved by a mixture of careful design of each component and special techniques for fault-tolerance [15].

4.1. One-way communication components

When the information is propagated from lower level to higher level systems, we need a one-way communication component that assures that this communication preserves the secure information flow.

Let us consider some requirements of this component.

1. **Confidentiality:** In an MLS system, the confidentiality of high information is enforced by disallowing information flow from high to low.
2. **Reliability:** One-way communication components should give assurance that the messages will be delivered. Message delivery can fail or be delayed due to (1) communication medium failure, or (2) the failure of the message receiver, etc. In general, reliable communication protocols are based on two-way communication. A sender sends a message and a receiver sends an acknowledgment (ACK) back to the sender. If the sender receives NAK or time-out, the sender re-sends the message. Reliability is an especially important issue in MLS systems because higher level receivers are not allowed to send acknowledgments (ACK or NAK) to lower level senders. Hence, if reliability and confidentiality requirements have to be satisfied simultaneously, controlled compromise must be the answer.
3. **Flow control:** The resources that are involved in delivery of messages may not always be available due to high traffic volume at certain periods of time. This component should provide a means for controlling the incoming traffic volume so that messages are not lost due to lack of resources.
4. **Fairness and availability:** This one-way component is, in general, a shared resource among many senders and receivers and should be shared fairly among senders. If a particular sender monopolizes a resource, so that other senders cannot use their fair share of resources, it is not only a fairness issue but also an availability (denial of service) issue. Hence, this component should be able to execute the system's fairness policy [13] and resist any potential misuse, including malicious denial of service attack.
5. **Performance:** Many one-way components intentionally delay the ACK time to reduce the covert channel capacity [6]. However, throughput and latency of this component may be as important as security. Hence, this component should have minimal impact on performance.
6. **Flexible implementation:** Distributed systems are usually dynamic. New members may be added to the system dynamically, and message traffic of the network is usually quite difficult to predict. Therefore, a one-way communication component should be flexible enough so that not only many different connection policies can be implemented but also resources can be re-allocated depending on dynamic workload.

Kang and Moskowitz introduced the basic NRL Pump as a device that balances these requirements [10, 12]. Note that no device can totally satisfy all requirements [18]. The Pump has been expanded [13] to deal with the network environment's added complications of fairness and denial of service. An abstract view of the Pump is shown in figure 6.

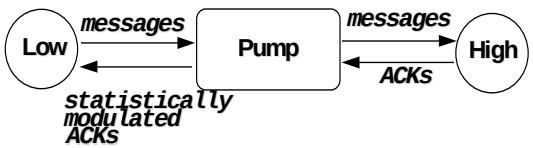

Figure 6: Basic Pump: The Pump with one sender and one receiver

The Pump establishes reliable and secure one-way communication between Low (a low level sender) and High (a high level receiver) by establishing a two-way reliable communication to Low and a two-way reliable communication to High. The Pump itself has to be reliable and recoverable to maintain reliable delivery service. Also, the ACKs to Low should be minimally affected by ACKs from High (otherwise, this ACK stream can be used to covertly pass information from High to Low). The Pump places a non-volatile buffer (size *n*) between Low and High, and gives ACKs at probabilistic times to Low based upon a moving average of past *m* High ACK times [10, 12]. A High ACK time is the time from when the Pump sends a message to High to the time when the Pump receives ACK from High. The probabilistic ACK to Low introduces noise into the Low ACK stream without degrading performance. If we consider the Pump to be at the high level, the ACKs from the Pump to Low violate the security policy[6]. However, the Pump is designed so that even though the security requirements are violated, the covert information leakage can be minimized while still satisfying the other requirements.

One can think of the Pump as a specialized network among low senders and high receivers. Hence, the Pump could be implemented as a (write-up) service of a MLS network (e.g., Boeing A1 MLS LAN [7]). The generalized Pump [14] can deliver messages from any sender to any receiver as long as the security policy is not violated. In this configuration, the role of sender and receiver is dynamically configured. Hence, a sender at one time can be a receiver at another time and vice versa. Note that such a dynamic role change is well suited for the DOC environment.

4.2. Cryptographic components

In general, commercial communication networks are insecure. Therefore, privacy, authenticity, and integrity of messages in the network are great concerns. Cryptographic techniques guard against such threats effectively. There are many known algorithms and protocols of this kind (i.e., encryption/decryption schemes, digital signatures, authentication systems) and ample literature is available. We therefore do not go into them in detail in this paper.

4.3. Multilevel secure (MLS) workstations

Some lower level information that is regularly used by higher level users/processes can be replicated from lower level to higher level systems. However, there is some unpredictable information that is needed by higher level users from time to time. One way to accommodate such needs is to use dedicated single-level workstations for lower level access (i.e., a high user may have more than one workstation). This is a very secure way to access lower level information.

However, if it is necessary to access several levels of information simultaneously or to copy and paste from one level to another level, then high assurance MLS workstations[7] can be used by the higher level users (thus a high user has only one MLS workstation). High level users who need to access lower

[6] Since the Pump is a trusted component, we can accept this limited security violation.

[7] Each organization that permits the use of multilevel workstations must accept the risk of potential information compromise.

level information can login to lower level systems through a lower level window on MLS workstations.

There are some efforts to implement trusted X-windows on TMACH[8]. However, we believe that there should be more effort in this direction. For example, an alternative approach to a two level workstation that is based on physical separation is given in figure 7.

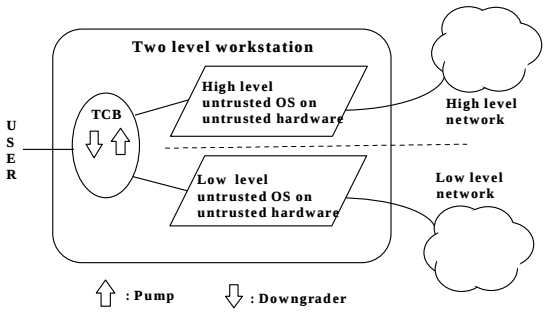

Figure 7: Simplified view of MLS workstations

All MLS workstation users are cleared to access information of the highest available level. An untrusted operating system exists on untrusted hardware for each security level. The TCB acts as a trusted switchboard that connects the user to the correct untrusted system based on the user's request. If the MLS workstation allows the *copy and paste* function from a low level window to a high level window, a Pump like one-way component may be needed. If it also allows the *copy and paste* function from a high level window to a low level window, then a downgrader component is necessary. Since this approach is based on physical separation, it may be easier to be evaluated/certified.

Since this type of MLS workstation can provide a capability to access (read and write) several levels, it may be used as a platform to execute trusted MLS applications [3].

4.4. Downgraders

Downgraders are necessary components but are also high risk components. It is well known that secret information can be encoded inside innocent looking images, e.g., [2]. Hence, human review may not be adequate to guarantee the prevention of information leakage. There are some downgraders known as *guards* (e.g., SAGE[9]). The use of such devices should be minimized because downgrading operations can potentially leak a large amount of information. One way to minimize the use of this device is to store information at the correct security level in the first place (i.e., there is a tendency to overclassify information and later downgrade it).

5. Architectural Solutions

Drawbacks of extending traditional MLS approaches to DOC have been discussed in section 2. In section 3, the needs of MLS users are described. In this section, we propose solutions to two configurations that were introduced in section 3. We also explain how the critical components that were introduced in section 4 can be used in our architectural framework. Our solution is based on the separation of security critical functions from non-security critical functions.

From configurations 1 and 2, the overall system requirements can be summarized as follows:

- High level users/processes need to frequently access a portion of low level information.
- The messages on a non-secure network need to be protected.
- Occasionally, there is some information that needs to be downgraded.
- There are a few high level users who need to occasionally browse low level information.

Recall that the system security policy can be expressed in terms of information flow:

No high level information should pass to lower level users/processes and lower level information should be available to higher level users/processes.

Our proposed solutions do not assume an MLS ORB due to reasons described in section 2.2. However, we want to take advantage of all the security services of CORBA or COM within a single level (hence, an ORB per security level). When lower level information or services are needed by higher level users, those can be shared through (1) replication of lower level information to higher level systems or (2) high level user's direct access to lower level systems through a dedicated or MLS workstation. Which information has to be replicated depends on system (security) design. If the need for lower level information by high level users/processes is known (predicted), then it should be replicated.

5.1. A secure architecture for configuration 1

Configuration 1 shows the need to connect system-high systems. Figure 8 shows our proposed solution to such needs.

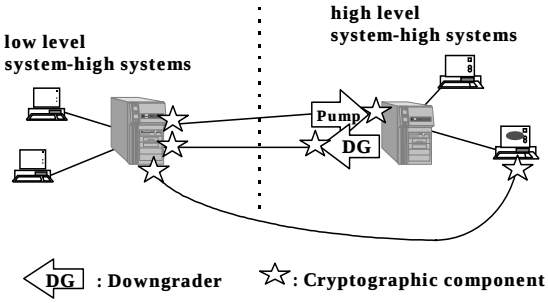

Figure 8: Architectural solution to configuration 1

The low information that is used frequently by high level systems can be replicated to high systems through the Pump. How often the replicas have to be updated can be determined by the type of information and the usage of information at the high level. The SINTRA MLS database system [9, 11] uses replication as a method to share lower level information with higher level systems. Also, there are several known algorithms that guarantee consistency between primary data and replicas [9, 20]. If the network is not physically secured then cryptographic components can be used to protect information on the network.

[8] TMACH is Trusted MACH operating system from TIS Inc.
[9] SAGE is a standard automated guard environment from WANG.

When high level users need to share sanitized versions of information with lower level users, they can do so through a downgrader. MLS workstations can be used when high level users need to browse lower level information. The low level portion of the MLS workstation is directly connected to the low level system/network. Hence, this operation does not violate the information security policy of the system. Note that the Pump and the downgrader are trusted high level components and should be protected accordingly.

5.2. A secure architecture for configuration 2

The proposed solution to configuration 2 (see figure 9) is similar to that of configuration 1 except that we now include a MLS network. In this case, the generalized Pump can be implemented as a write-up (one-way) service for the MLS network.

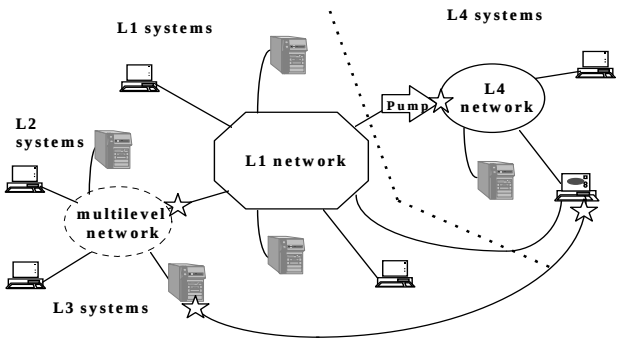

Figure 9: Architectural solution to configuration 2

Note that we did not present a complete solution in the figure due to cluttering -- only a few critical components are inserted between systems where we wish to be specific. For example, since the generalized Pump is a service of an MLS network, we do not show this component in figure 9; however, it is implicitly there. Of course, depending upon the actual security requirements, the configuration might require additional security measures (e.g., link encryption, prevention of traffic analysis).

5.3. Design review

In section 3.3, we specified functional requirements. Let us review how the proposed architectural solutions meet the requirements.

- ***Information flow from low to high system.*** If lower level information needs to be sent to higher level systems: (1) the information is encrypted and authenticated (if necessary), (2) it is sent to the Pump through a network, (3) it is delivered to the final destination, and (4) it is decrypted and verified (if necessary). Note that steps (3) and (4) can be reversed depending on system requirements (see [16] for covert channel analysis).
- ***Privacy, integrity, authenticity of information.*** If information travels through an unprotected portion of network and the information needs protection, then cryptographic components can be used.
- ***Higher level users may need to access lower level resources.*** If higher level users need to access lower level information that has not been replicated to a higher level system, then a higher level user can login to lower level system through a MLS workstation. If the network is not protected and the information requires protection then cryptographic techniques should be used.
- ***Availability of resources.*** No single technique can solve this problem, although the fault-tolerant community uses replication to increase availability [15]. Our proposed architectures use replication as a way to share lower level information with higher level processes/users. We believe that smart replication engineering can help to achieve the goals of availability, performance, and sharing with minimal replication.
- ***Downgrading.*** If there is a need to downgrade information then a downgrader should be used (see figure 8). If the downgraded information is still at a higher security level than the security level of the unprotected portion of the network, then cryptographic techniques should be used.

Additional benefits of our approach are as follows:

- ***Reduced cost.*** The overall cost of our approach will be much lower than that of the naive extensions of the traditional MLS approach because our approach encourages the use of commercially available products.
- ***Provision of a migration path for legacy systems.*** Legacy systems can participate in new federations without jeopardizing security because these systems are isolated by security critical components.
- ***Provision of a migration path to new technologies.*** When new products or technologies are available, an organization can incorporate these in the federation without affecting other organization/systems. This is true because systems from different organizations are strongly separated by security critical components.

Promotion of sharing, security, and autonomy. Since the security of our proposed approach is flexible and easy to understand, it encourages organizations to participate in federation, and while retaining full control of their own systems. Each organization can decide which critical components are needed, depending on their own security and functionality needs.

6. Conclusions

Users of MLS technology want to take advantage of new technologies. In general, incorporating new technology into MLS products is slower than that to non-secure products due to the rigorous and long evaluation and certification processes.

In this paper, we proposed an architectural framework that can speed up the process of introducing new technologies (e.g., distributed object-oriented computing) to the MLS community. We examined the drawbacks of traditional MLS approaches and took a fresh look at the needs of the MLS community. We then introduced critical components that can facilitate the needs of the MLS community. Since these components must go through a rigorous evaluation and certification processes, they should be as independent as possible from non-critical technology. We then introduced architectural solutions to the needs of the MLS community. The proposed solution is based on a “separation of concerns” principle: the security critical components should be separated from non-security components. We believe the proposed approach reduces cost, provides a migration path for

legacy systems and to new technologies, and promotes information sharing while maintaining the security and autonomy of organizations.

Our architectural solution provides a framework to share information securely. In our framework, the operational security policy is based on information flow among different security levels. Under the framework that we have discussed, more research is needed to make a secure federation acceptable to MLS users. One such research area is data security engineering: how to organize data to make the sharing as smooth as possible with minimal overhead.

Acknowledgments

We thank Steven Greenwald and Ruth Heilizer for their helpful comments.

References

1. Bell, D. E. and LaPadula, L. J. "Secure computer system: Unified exposition and multics interpretation," The Mitre Corp. 1976.
2. Cha, S. D., Park, G. H., and Lee, H. K. "A solution to the on-line downgrading problem," Proceedings of 11th Computer Security Applications Conference, pp. 108 - 112, New Orleans, LA, 1995.
3. Costich, O. and Kang, M. H. "Maintaining multilevel transaction atomicity in MLS database systems with replicated architecture," Proceedings of 7th Annual IFIP WG11.3 Working Conference on Database Security, pp. 216 - 240, Huntsville, AL, 1993.
4. Department of Defense, "Trusted computer system evaluation criteria," DoD5200.28-STD, 1985.
5. Froscher, J. N., Golschlag, D. M., Kang, M. H., Landwehr, C. E., Moore, A. P., Moskowitz, I. S., and Payne, C. N. "Improving inter-enclave information flow for a secure strike planning application," Proceedings of 11th Computer Security Applications Conference, pp. 89 - 98, New Orleans, LA, 1995.
6. Hu, W. M. "Reducing timing channels with fuzzy time," Proceedings of IEEE Symposium on Security and Privacy, pp. 8 - 20, Oakland, CA. 1991.
7. Janeri, J. V., Darby, D. B., and Schnackenberg, D. D., "Building higher resolution synthetic clocks for signaling in covert timing channels," Proceedings of IEEE Computer Security Foundations Workshop, pp. 85 - 95, Ireland, 1995.
8. Jensen, C., *et. al*. "SDDM: A prototype of a distributed architecture for database security," Proceedings of Conference on Data Engineering, 1989.
9. Kang, M. H., Froscher, J. N., and Costich, O. "A practical transaction model and untrusted transaction manager for multilevel-secure database systems," Proceedings of 6th Annual IFIP WG11.3 Working Conference on Database Security, pp. 289 - 310, 1992.
10. Kang, M. H. and Moskowitz, I. S. "A Pump for rapid, reliable, secure communication," Proceedings of ACM Conference on Computer & Communication Security, pp. 119 - 129, Fairfax, VA, 1993.
11. Kang, M. H., Froscher, J. N., McDermott, J., Costich, O., and Peyton, R. "Achieving database security through data replication: The SINTRA prototype," Proceedings of 17th National Computer Security Conference, pp. 77 - 87, Baltimore, MD. 1994.
12. Kang, M. H. and Moskowitz, I. S. "A data Pump for communication," submitted for publication, also available as NRL Memo. Report 5540-95-7771, 1995.
13. Kang, M. H., Moskowitz, I. S.. and Lee, D. C. "A network Pump," IEEE Transactions on Software Engineering, vol. 22, no. 5, pp. 329 - 338, 1996.
14. Kang, M. H. and Moskowitz, I. S. "A generalized Pump," In preparation.
15. Maffeis, S. "Adding group communication and fault-tolerance to CORBA," Proceedings of USENIX Conference on Object-Oriented Technologies, Monterey, CA, 1995.
16. Meadow, C. and Moskowitz, I. S. "Covert channels - A context based view," Proceedings of the Workshop on Information Hiding, Cambridge, UK, 1996.
17. McLean, J. D. "A general theory of composition for trace sets closed under selective interleaving functions." Proceedings of 1994 IEEE Computer Society Symposium on Research in Security and Privacy, pp. 79 - 93, Oakland, CA, 1994.
18. Moskowitz, I. S. and Kang, M. H. "Covert channels --- Here to stay?," Proceedings of COMPASS '94, pp. 235 - 243, Gaithersburgs, MD, 1994.
19. Object Management Group "CORBA Security," OMG document 95-12-1, 1995.
20. Zhang, A. and Elmagarmid, A. K. "A theory of global control in multidatabase systems," The VLDB Journal, pp. 331 - 360, vol. 2, No. 3, July 1993.